OCR Enhancement for Ancient Indic Manuscripts via Hybrid Image Processing and CNN-Based Script Classification

Nehal Mantri
Department SCOPE
Vellore Institute of Technology
Chennai Campus

Aditya Mukati
Department SCOPE
Vellore Institute of Technology
Chennai Campus

Nikhil Joshi
Department SCOPE
Vellore Institute of Technology
Chennai Campus

Geetha S.
Department SCOPE
Vellore Institute of Technology
Chennai Campus

***Abstract*—Ancient Tamil and Sanskrit manuscripts face degradation and optical character recognition (OCR) challenges due to script complexity, ink bleed, and uneven illumination. To address these challenges, we propose an end-to-end pipeline combining image restoration, language classification, and OCR. A seven-stage preprocessing workflow (grayscale conversion, Contrast-Limited Histogram Equalization, upscaling using ESRGAN, non-local means filtering, etc) enhances image quality, reducing OCR Character Error Rates (CER). A custom Convolution Neural Network (CNN) was trained on a novel dataset of 246k segmented word images of both Sanskrit and Tamil. It achieved 98% accuracy in classifying Sanskrit/Tamil scripts via majority voting, which enables language-specific OCR optimization for better accuracy. The preprocessed images are then fed to the Google Vision API for text extraction, and the OCR output is processed by a prompt-engineered DeepSeek-R1 model for translation and summarization of text. Evaluated on 50 pages of Yogini Tantra, the pipeline demonstrates a 55% CER improvement over raw-image OCR, as the OCR Character Error Rates (CER) decreased from 0.20 to 0.09.**

***Keywords*—Image Processing, CNN, Optical Character Recognition (OCR), Language Classification, Word Segmentation**

I. Introduction

The preservation of ancient Tamil and Sanskrit manuscripts is critical to safeguarding South Asia's literary, philosophical, and scientific heritage. Sanskrit traces its origins back to Vedic Sanskrit, the language of the Vedas, while there are many epic Tamil scripts like Thirukural and Purananool. However, these manuscripts suffer from environmental factors such as ink bleed, fungal growth, natural deterioration and physical wear [1]. Furthermore, challenges such as similarities in letter forms, script complexity, non-standardized representations, and a large symbol set make the task of Sanskrit and Tamil character recognition from manuscript images particularly difficult [2], resulting in high error rates in unprocessed images.

Our work addresses these issues with an end-to-end pipeline that integrates adaptive image preprocessing, CNN-based language classification, and language-specific OCR. Image processing technology can decontaminate, enhance and restore ancient books to make their contents more legible and readable, facilitating research and utilization[3]. Therefore, an eight-stage preprocessing technique is employed to enhance manuscripts for OCR, involving grayscale conversion, wavelet-based denoising, Contrast-Limited Adaptive Histogram Equalization (CLAHE), super-resolution enhancement using ESRGAN, adaptive background subtraction, thresholding using the Sauvola method, unsharp masking, and finally, median filtering to remove minute salt-and-pepper noise. A custom Convolution Neural Networks (CNN) architecture with four Conv2D layers, trained on a novel dataset of 246,000 segmented Tamil and Sanskrit word images curated from 2,765 manuscript pages, achieves over 99% in classifying a word as Tamil or Sanskrit. To classify the language of a manuscript page, the page is segmented into individual words. Our CNN model predicts the language (Tamil or Sanskrit) for each word, and the final manuscript classification is determined via majority voting. After this classification, we are able to extract text using language-specific Tesseract OCR engine.

Evaluated on 50 scanned pages of the *Yogini Tantra*, a 16th-century Sanskrit manuscript, the pipeline reduces OCR character error rates (CER) by 55% (0.20 to 0.09) compared to raw-image processing. This improvement offers a scalable solution for mass digitization. The curated dataset and our architecture address longing gaps in Indic manuscript preservation, which will also enable future extensions to related languages like Malayalam and Telugu.

II. Literature Survey

Zhou et al. [3] explored the role of image processing technologies in the protection and digital restoration of ancient books. The study outlines a structured methodology involving high-precision scanning, enhancement, denoising, binarization, layout analysis, and text recognition to transform ancient texts into high-quality digital formats. These techniques significantly improve readability, accessibility, and the overall visual quality of deteriorated materials.The paper advocates for ongoing research and technological development to further refine these methods and support cultural heritage conservation through digitization.

Regional script digitization faces unique hurdles, as seen in Jayashree et al.'s work on Tulu palm-leaf manuscripts [4].

Their study addresses the scarcity of efficient OCR solutions by employing various machine algorithms, achieving highest accuracy of 92.3% with the random forest algorithm. Their classifier-level fusion approach played a key role in enhancing recognition accuracy, especially for the complex Tulu characters with varying font styles and sizes.

Alkhazraji, Baheeja & Alzubaidi [5] conducted a literature review to examine the different approaches and mthods used for restoring ancient texts that are often damaged over time due to environmental factors. They reviewed various works which used deep learning methods. Some of these were identifying Arabic handwritten characters using CNN [6], restoring ancient Chinese characters with dual generative adversarial networks (GANs) [7], rotation and reflection-modified PixelCNN to restore letters in ancient Qumran writings [8], recognizing old Chinese characters on GANs [9], text extraction and restoration of old handwritten documents using computer vision and machine learning methods. [10]. Finally they concluded that despite researchers solved multiple obstacles that traditional restoration methods faced, there is still space for development. Future works could concentrate on improving improving deep neural network performance by adding domain-specific knowledge and creating more complicated models that can adapt to varied writing styles and languages.

Yuadi et. al [1] explored five types of filters, Gaussian Blur, Adaptive Thresholding, Canny Edge Detection, Morphology, and Sobel Algorithm, to address the readability challenges of ancient Javanese manuscripts. Their study evaluated how each filter enhances visual clarity while maintaining structural integrity. For this, they used evaluation metrics such as the Structural Similarity Index Measure (SSIM) and Normalized Cross-Correlation (NCC). They concluded that Gaussian Blur supports visual authenticity and segmentation tasks, Adaptive Thresholding and Morphology are effective for improving legibility, especially for OCR, while Canny Edge and Sobel are useful for detecting text boundaries or damage, though less suitable for full preservation.

Borah, Vijapur & Kumar [11] dealt with the challenges digitizing ancient Kannada manuscripts. Their main focus was on development of an Optical Character Recognition (OCR) model which can decipher Kannada characters and convert them into digital characters. Their method employs CNN algorithm which accepts pre-processed data and predicts each character based on the trained and tested database. Preprocessing steps include Grayscale conversion, Denoising and Do Contour detection. These methods increased the accuracy by attaining the optimal performance.

Menon, Sreekumar and Nair [12] also worked upon increasing tesseract's OCR accuracy by using adapti ve Gaussian thresholding and pre-processing techniques like binarization to improve the quality of the images.They carried out word-level and line-level segmentation utilizing the blob approach and regional zoning. Lastly they recognized characters and created the database. They got a recognition accuracy of above 93% which demonstrated successful identification and digitization of Malayalam manuscripts.

Moudgi et al. [13] developed an OCR which was able to read Devanagiri. They used AlexNet transfer learning model for this classification. They utilized a Devanagari manuscript data set consisting of 7536 characters. They obtained a maximum accuracy of 95.4

Alexander et al. [14] investigated image restoration techniques for palm leaf manuscripts. The study utilized several pre-processing techniques, including Cellular Automata (CA)-based optimal edge detection, Otsu's algorithm, and the Hybrid Isotropic Diffusion with Perona–Malik (HIDPM) model, to restore the manuscripts and improve their legibility for character recognition. The paper found that the CE-based edge detection technique outperformed standard edge detectors in terms of performance, while the HIDPM model provided the best results, offering accurate image restoration with low execution time

III. IMAGE PREPROCESSING

Image preprocessing begins with a seven-stage workflow to enhance manuscript. We will use an ancient degraded Sanskrit scripture to illustrate output of each stage.

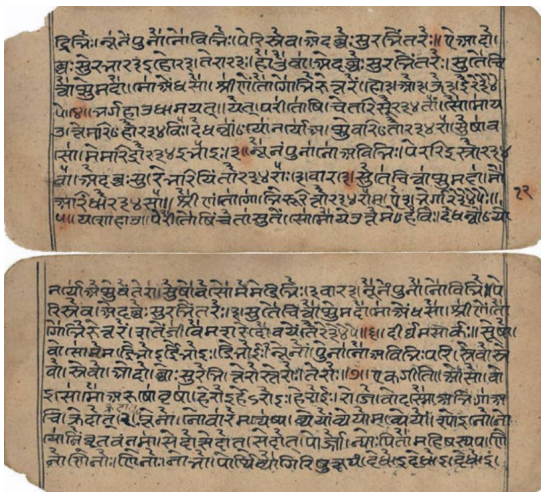

Fig. 1. Ancient Sanskrit scripture to be used for demonstration

A. Grayscale Processing

The first step in the pipeline involves converting the input color image to grayscale. This reduces the complexity of the data by focusing only on luminance, which is more relevant for extracting textual content. Ancient manuscripts often have color variations due to staining or ink oxidation[1], which can mislead enhancement algorithms. Grayscale conversion ensures that the rest of the enhancement techniques operate on a uniform and simplified representation of the image. Each

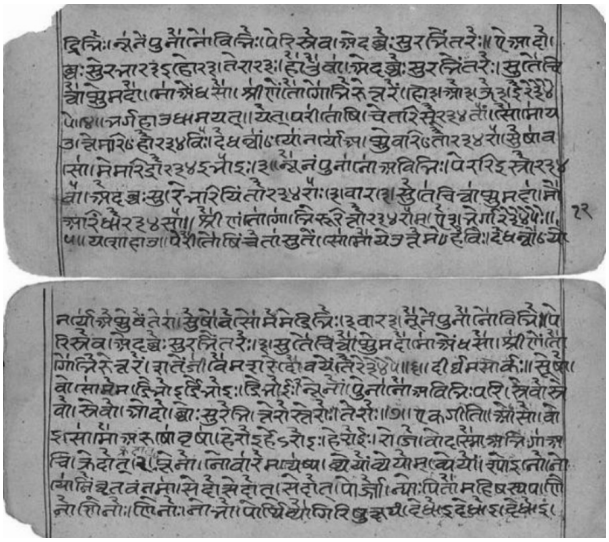

Fig. 2. Grayscale Conversion

pixel in the image, initially characterized by three color components—Red (R), Green (G), and Blue (B)—is transformed into a single grayscale intensity[1]. The weighted value of the three channels in the color image is taken as the gray value:

$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \tag{1}$$

B. Wavelet-Based Denoising

After the image is converted to grayscale, the next priority is to reduce subtle noise while also retaining the stroke sharpness. Historical documents have background noise due to continuous degradation over time, to address this, wavelet-based denoising is applied. Wavelet denoising operates by transforming the signal into the wavelet domain, where the signal's essential features are represented by a few large-magnitude coefficients, while noise tends to be distributed among many small-magnitude coefficients. By applying thresholding techniques such as soft thresholding with BayesShrink, small coefficients associated with noise are suppressed, and the signal is reconstructed using the inverse wavelet transform.

BayesShrink Soft Threshold:

$$\hat{w} = \begin{cases} \text{sign}(w) \cdot (|w| - T), & \text{if } |w| > T \ 0, \\ \text{otherwise} \end{cases} \tag{2}$$

where

- w are the noisy wavelet coefficients,
- $\hat{w}$ are the denoised wavelet coefficients,
- $T = \frac{\sigma^2}{\sigma_x}$ is the BayesShrink threshold,
- σ is the noise standard deviation,
- σ_x is the standard deviation of the noiseless signal coefficients.

This approach allows for the attenuation of subtle noise while preserving important signal features.Traditional filtering methods, like low-pass filters can also remove noise but often tend to blur edges or other features. Hence this approach helps in effective suppression of noise while retaining delicate script contours which is essential for text recognition and extraction.

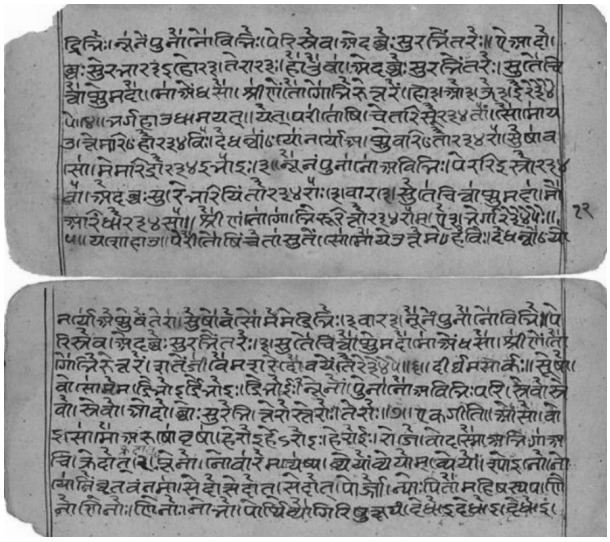

Fig. 3. Wavelet Denoising

C. Local Contrast Enhancement via CLAHE

After wavelet denoising, Local contrast enhancement using CLAHE (Contrast Limited Adaptive Histogram Equalization) is applied to highlight faint text without amplifying residual noise. This step is most effective after denoising because it relies on clean pixel intensity differences to enhance contrast selectively in faded or uneven regions, something especially useful for manuscripts with patchy ink or inconsistent lighting. Traditional histogram equalization may over-amplify noise or decrease clarity of fine details, especially in non-uniform regions. CLAHE avoids this by performing contrast enhancement within localized tiles, redistributing intensities to highlight faded text without distorting background uniformity.

CLAHE divides the image into non-overlapping regions called tiles (e.g., 4×4). For each tile, a histogram of pixel intensities is computed and then clipped at a predefined clip limit $\mathcal{C}$ to prevent over-amplification of noise in relatively uniform regions. The clipped histogram is used to compute a transformation function $T(i)$, which maps the input pixel intensity i to an output intensity using the cumulative distribution function (CDF):

$$T(i) = \frac{(N-1)}{M \times N} \sum_{j=0}^{i} h(j) \tag{3}$$

where:

- $T(i)$ is the output intensity corresponding to input intensity i,
- $h(j)$ is the clipped histogram value for intensity level j,
- $M \times N$ is the total number of pixels in the tile.

The enhanced tiles are combined using bilinear interpolation to avoid boundary artifacts between adjacent tiles.

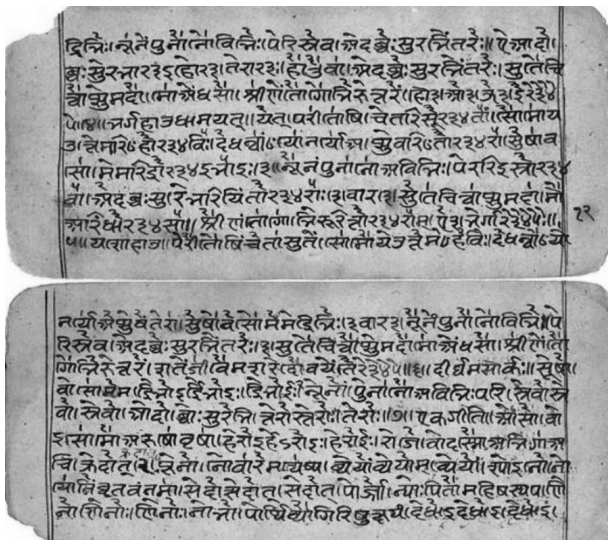

Fig. 4. CLAHE Enhanced

D. Super-Resolution Enhancement Using ESRGAN

Following contrast improvement, the resolution of the manuscript image is enhanced using ESRGAN (Enhanced Super-Resolution Generative Adversarial Network). Since text clarity is crucial, particularly for complex scripts, this deep-learning-based technique helps reconstruct lost details and sharpens edges. It's applied after contrast enhancement to ensure that both structure and subtle textual variations are visible and well-defined. This step significantly enhances character visibility, especially for cursive or degraded scripts, improving the potential for OCR.

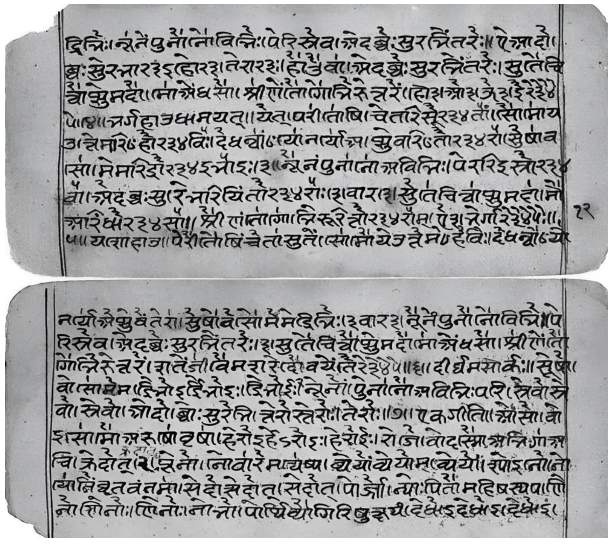

Fig. 5. ESRGAN Enhanced

E. Adaptive Background Subtraction

Once the text is clearer and sharper, the next goal is to handle irregular backgrounds caused by aging artifacts like paper wrinkles, stains, or shadows. Adaptive background subtraction is introduced here to normalize the lighting and bring uniformity to the background. A blurred approximation of the background is generated using a large-kernel Gaussian filter, which is then subtracted from the original image through pixel-wise division. This operation normalizes illumination across the manuscript, effectively highlighting the textual foreground while minimizing the influence of stains and aged paper textures.

$$I_{\text{enhanced}}(x,y) = \left(\frac{I_{\text{original}}(x,y)}{G_\sigma * I_{\text{original}}(x,y)} \right) \times 255 \qquad (4)$$

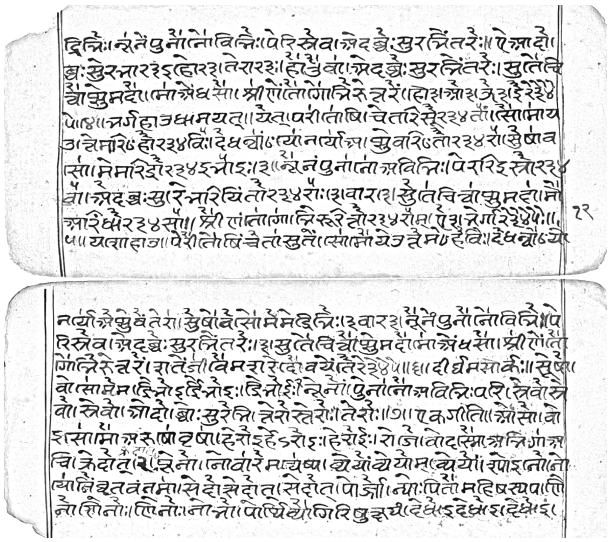

Fig. 6. Adaptive Background Subtraction

F. Sauvola Adaptive Binarization

Now that the background has been normalized, the manuscript is ready for binarization, a crucial step that determines how well the text stands out from its surroundings. In this process, we use Sauvola's adaptive thresholding method, which is particularly well-suited for historical documents. Unlike global methods that apply a single threshold across the entire image, Sauvola's approach adapts to the local mean and standard deviation of pixel values within small regions. This makes it incredibly effective in handling non-uniform lighting, faded ink, and text over stains or textures which are all common characteristics of aged manuscripts.

By computing the threshold locally, the method ensures that even faint or partially degraded characters are retained, rather than lost in the noise. This is especially important in historical texts, where every subtle stroke may carry significant linguistic or cultural meaning. Moreover, Sauvola's algorithm is less likely to mistake background blemishes for text because of its sensitivity to local contrast. The result is a high-contrast binary image that maintains the fine details of the script while cleanly separating it from background artifacts.

$$T(x,y) = m(x,y) \left[1 + k \left(\frac{s(x,y)}{R} - 1 \right) \right] \qquad (5)$$

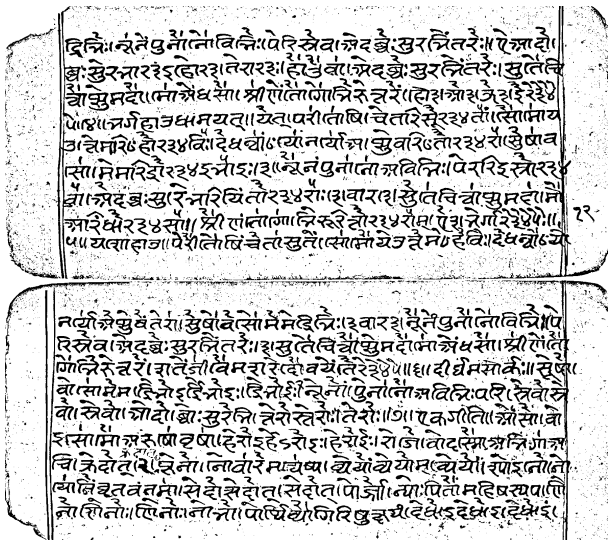

Fig. 7. Savuola Output

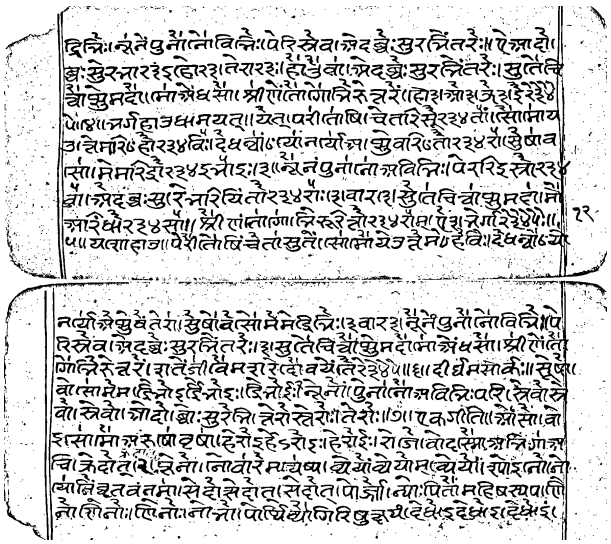

Fig. 8. Unsharp Masking

where:

- $T(x, y)$: Sauvola threshold at pixel (x, y)
- $m(x, y)$: Local mean intensity (35×35 window)
- $s(x, y)$: Local standard deviation (35×35 window)
- R: Dynamic range of standard deviation ($R = 128$ for 8-bit images)
- k: Sensitivity parameter ($k = 0.10$ in implementation)

Binarization:

$$I_{\text{binary}}(x, y) = \begin{cases} 255 & \text{if } I_{\text{original}}(x, y) > T(x, y) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

G. Unsharp Masking and Morphological Refinement

After binarization, some text may still have fuzzy edges or broken strokes. To fix this, unsharp masking is applied to enhance edge details, followed by morphological closing operations using a small kernel to reconnect fragmented parts of the text. Performing this step after binarization ensures that these enhancements are applied directly to the foreground text without affecting the background. This step is essential for ensuring the structural integrity of characters, particularly those with broken or fragmented forms due to ink decay.

$$I_{\text{sharpened}} = \alpha I_{\text{original}} - \beta G_\sigma * I_{\text{original}} \tag{7}$$

where:

- I_{original}: Input grayscale image
- $G_\sigma * I_{\text{original}}$: Gaussian-blurred image ($\sigma = 3$, kernel auto-calculated)
- $\alpha = 1.5$, $\beta = 0.5$: Sharpening coefficients
- $\sigma = 3$: Controls blur strength for edge extraction

Post-sharpening morphological closure:

$$I_{\text{final}} = (I_{\text{sharpened}} \oplus K) \ominus K \tag{8}$$

where:

- K: 2×2 rectangular structuring element
- $\oplus$: Dilation operator
- $\ominus$: Erosion operator

H. Median Filtering

As the final touch, median filtering is used to clean up any leftover noise, such as speckles or binarization artifacts. This non-linear filter preserves the edges of the text while removing small, isolated distortions (unlike mean filters). This step is positioned at the end of the pipeline and ensures the output image clear, denoised and readable and ready for further tasks like language classification and OCR.

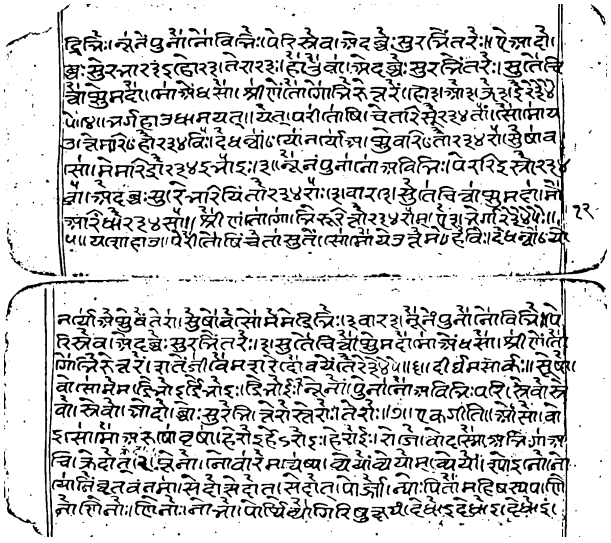

Fig. 9. Median Filtered Output

The Median Filter is calculated as:

$$I_{\text{filtered}}(x, y) = \underset{(i,j)\in\Omega}{\text{median}} \left(I_{\text{original}}(x + i, y + j) \right) \tag{9}$$

where:

- I_{original}: Input grayscale image (noise-corrupted)
- Ω: $k \times k$ neighborhood around pixel (x, y)
- k: Kernel size ($k = 5$ in implementation)
- median: Median operator over the $k \times k$ window

IV. METHOD

A. DATASET CREATION

Since we couldn't find a satisfactory dataset for language classification, we created a custom corpus of word images. Source materials were procured from the Digital Library of India , a public-domain repository, including Tamil manuscripts (Devi-Khadgamala, Ozhiviloodukkam, Changga-Ilakkiyam, Kannappa-Nayanar, Tirupuri-Mutal, and Sri-Vachanapuushhand) and Sanskrit manuscripts (Niruktam-Vol-II, Margpuri-Sudhi, Shyama-rahasya, Shivaji-Nibandhavali, and Panchapadika-vivarana). Individual pages were extracted from these documents, resulting in a collection of over 2765 page images.

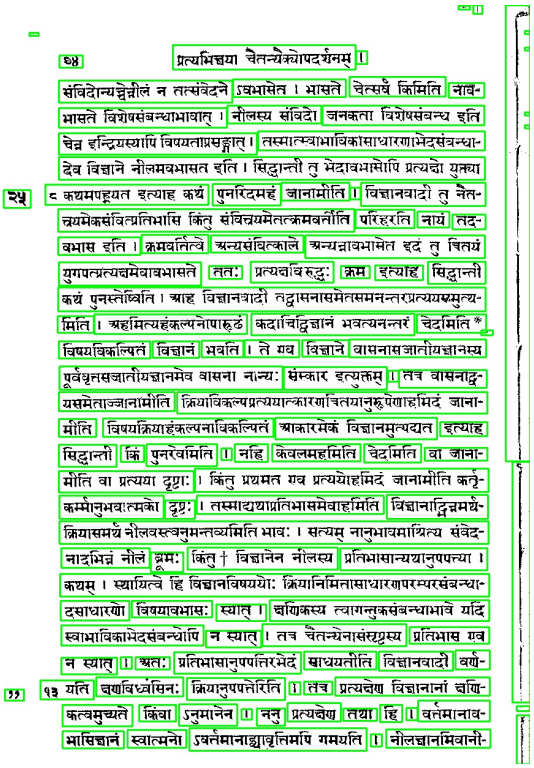

Fig. 10. Sample Sanskrit Manuscript

Then, we isolated individual words from these full pages using word segmentation and got over 246k words (126k Sanskrit words and 120k Tamil words). Images were normalized to 64x196 pixels grayscale for training purposes. The dataset was split in 80-20 ratio to generate 197,280 training and 49,320 validation samples.

B. CNN Training

The proposed convolution neural network (CNN) for language classification consists of four hierarchical convolutional

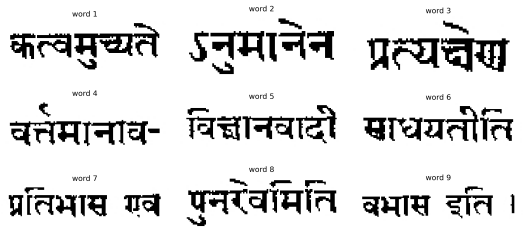

Fig. 11. Sample Sanskrit Words

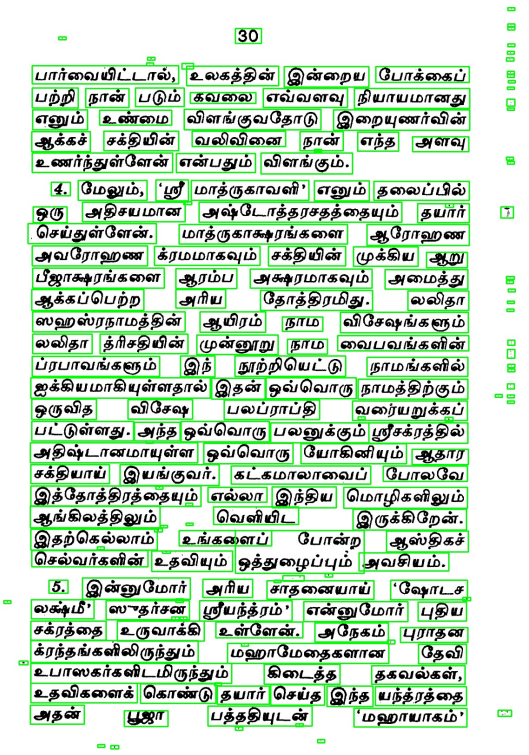

Fig. 12. Sample Tamil Manuscript

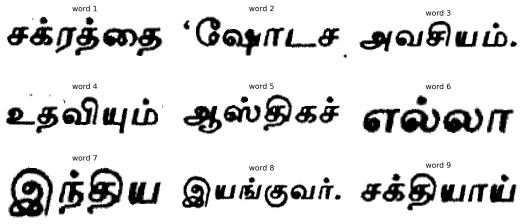

Fig. 13. Sample Tamil Words

blocks followed by dense layers. The architecture begins with a 32-filter 5×5 convolutional layer with ReLU activation and same padding, followed by batch normalization and 2×2 max-pooling to prioritize local stroke pattern extraction. The convolution blocks employ 3x3 kernels to incrementally capture language-specific features, leveraging the morphological differences between Tamil and Sanskrit. Early layers detect low-level curvature prevalent in Tamil characters (e.g., rounded strokes ‘’ or ‘’), while deeper layers identify rectilinear conjuncts characteristic of Sanskrit (e.g., the angular ‘’ or ‘’). This progressive feature extraction is enhanced through batch normalization, which stabilizes activation distributions across (64,128,256) filter depths, and max-pooling, which preserves spatial relationships. Global average pooling is used to reduce parameter dimensionality before feeding into a 256-unit dense layer with 50% dropout for regularization. A final softmax layer outputs Tamil/Sanskrit class probabilities. The model was trained for 30 epochs (early stopping enabled) using the Adam optimizer (=0.001) with a batch size of 32 images and categorical cross-entropy loss, with dynamic learning rate reduction (factor=0.5 after 3 validation loss plateaus).

C. Language Classification

After the preprocessing is done, the output image is now ready for language classification. A majority voting mechanism on word-level predictions is done to determine the language of each manuscript. The word segmentation is achieved by: Individual words segmented from the preprocessed images are fed into the trained CNN model , which outputs Tamil/Sanskrit probabilities for each word based on the learned morphological features. A majority vote assigns the final language label of the manuscript. Fig. 14 shows a manuscript being classified as Sanskrit with majority vote: ’sanskrit’: 772, ’tamil’: 41. Blue coloured boxes signify words being classified as Sanskrit while Red colored boxes shows Tamil classified words.

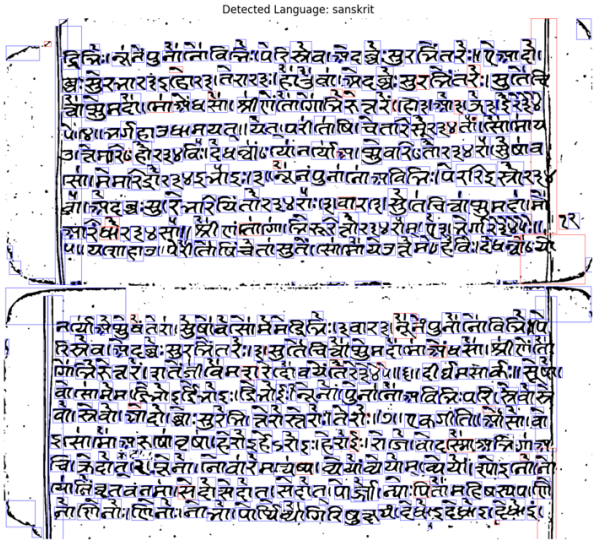

Fig. 14. A Sanskrit image being classified as Sanskrit

After language detection, the manuscript goes through language-specific optical character recognition (OCR) using Tesseract.

V. RESULT

A. Language Classification Model

To evaluate the CNN model, it was tested upon 49,320 validation samples (20% of the 246,000-word dataset).

1) Classification Accuracy: The training and validation performance of the custom Convolutional Neural Network (CNN) designed for classifying words in Sanskrit and Tamil is illustrated in Fig. 15. The left subplot demonstrates the classification accuracy across training epochs, while the right subplot shows the corresponding loss values. The training accuracy rapidly increased initially, reaching 99% in just 2 epochs. The validatin accuracy showed similar trend by instantly reaching 98% and maintaing a stable performance throughout the training process. The model achieved, hence achieved a maximum of 99.06% validation accuracy, closely aligning with training accuracy (99.71%). This shows that the model performs very well on unseen data.

As we can see in the right subplot, the training loss decreased sharply in the early epochs, reaching a value closer to zero. The validation loss also decreased initially, which demonstrates the model’s ability to minimize errors on the validation set. There is also a temporary increase in validation loss which could potentially be due to a slight overfitting to specific training examples during that particular epoch.

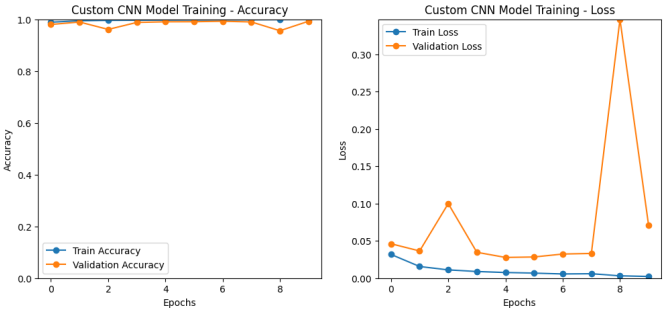

Fig. 15. Training and Validation Accuracy of CNN

2) Confusion Matrix: As shown in Fig. 16, the model exhibits near-symmetric performance across Tamil and Sanskrit. The model achieved a high accuracy in classifying. Out of a total of 24,079 Sanskrit words, the model correctly classified 23,990, with only 89 words misclassified as Tamil. Similarly, for Tamil words, the model correctly identified 24,875 out of 25,252 words, misclassifying only 377 as Sanskrit. This indicate that the CNN model is efficient in distinguishing between the two languages. The high values across the diagonals also gives a positive result on model’s performance.

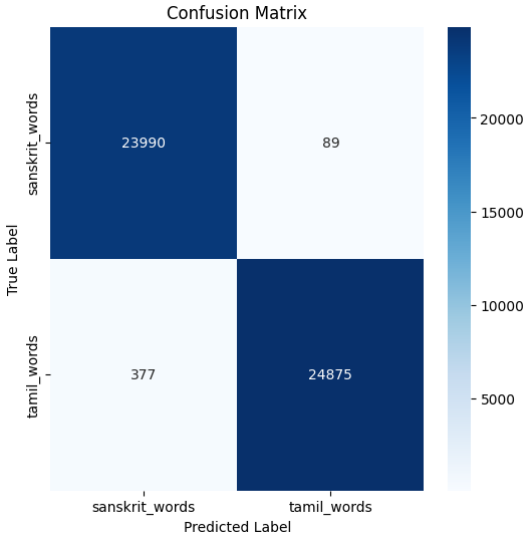

Fig. 16. Confusion Matrix

Fig. 17. Prediction label of sample images

B. Classification Report

We also calculated model's performance, precision, recall, and F1-score for each language, as summarized in Table 1. The model achieved a precision of 0.98 for Sanskrit words, which indicated that the model 98% correct whenever it a predicted word was Sanskrit. The recall of 1.00 for Sanskrit means that the model correctly identified all Sanskrit words in the test set. Similarly, for Tamil words, the model demonstrated an excellent performance with a precision of 1.00, indicating that all words predicted as Tamil were Tamil. The recall of 0.99 for Tamil showed that the model correctly identified 99% of the Tamil words.

The macro average F1-score across both classes was 0.99, and the weighted average F1-score, which accounts for the class imbalance (slightly more Tamil words in the test set), was also 0.99, consistent with the macro average. These high precision, recall, and F1-score values further confirm the effectiveness of the custom CNN model in accurately classifying words from Sanskrit and Tamil.

TABLE I
CLASSIFICATION REPORT FOR TAMIL/SANSKRIT WORD RECOGNITION

Class	Precision	Recall	F1-Score	Support
Sanskrit	0.98	1.00	0.99	24,079
Tamil	1.00	0.99	0.99	25,252
Macro Avg	0.99	0.99	0.99	49,331
Weighted Avg	0.99	0.99	0.99	49,331

1) Prediction of Test Images: Fig. 17 illustrates the CNN's predictions on nine randomly sampled test images from the validation set.

C. OCR Performance

To evaluate our preprocessing pipeline, we performed OCR on 50 pages of *Yogini Tantra* which constituted of a total of 58,882 characters. OCR was done on the images before and after preprocessing and the result was compared using Character Error Recognition (CER) as illustrated in Fig. 18. The original images had a CER value of 0.09, while the preprocessed images achieved a lower CER of around 0.205. CER drastically improving by 50% (from 0.20 to 0.09) after preprocessing of images, showcases the effect of pipeline on OCR performance.

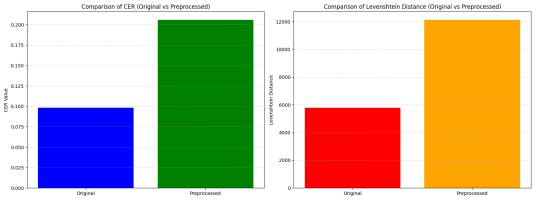

Fig. 18. CER comparison

REFERENCES

[1] I. Yuadi, N. Yudistra, H. Habiddin, and K. Nisa, "A Comparative Study of Image Processing Techniques for Javanese Ancient Manuscripts Enhancement", vol. 13, pp.36845 - 36857, 25 February 2025

[2] B. Kataria and H. Jethva, "Optical Character Recognition Of Sanskrit Manuscripts Using Convolution Neural Networks", vol. 18(5), pp.403-424, January 2021

[3] C. Zhou, M. Sun, C. Hao and Y. Chi, "Application and Exploration of Image Processing Technology in Ancient Books Protection and Design", 2024 5th International Conference on Computer Vision, Image and Deep Learning (CVIDL), April 2024

[4] J. K. R., S. Sinchana, M. K., D. Deekshitha, S. K. Sudarshan, and P. Kannadaguli, "Tulu Manuscript OCR: Preserving Ancient Wisdom through Character Recognition," in Proc. 2024 Second International Conference on Data Science and Information System (ICDSIS), Mangalore, India, 2024;

[5] A. Alkhazraji, K. Baheeja, and A. M. N. Alzubaidi, "Ancient Textual Restoration Using Deep Neural Networks: A Literature Review," in Proc. 2023 Al-Sadiq Int. Conf. Commun. Inf. Technol. (AICCIT), Karbala, Iraq, 2023; IEEE, pp. 64–65

[6] A. El-Sawy, M. Loey, and H. El-Bakry, "Arabic handwritten characters recognition using convolutional neural network," WSEAS Transactions on Computer Research, vol. 5, no. 1, pp. 11-19, 2017.

[7] B. Su et al., "A restoration method using dual generate adversarial networks for Chinese ancient characters," Visual Informatics, vol. 6, no. 1, pp. 26-34, 2022

[8] L. Uzan, N. Dershowitz, and L. Wolf, "Qumran letter restoration by rotation and reflection modified PixelCNN," in 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 23-29, Nov. 2017.

[9] J. Cai, L. Peng, Y. Tang, C. Liu, and P. Li, "TH-GAN: generative adversarial network based transfer learning for historical Chinese character recognition," in 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 178-183, Sept. 2019.

[10] M. Wadhwani et al., "Text extraction and restoration of old handwritten documents," Digital Techniques for Heritage Presentation and Preservation, pp. 109-132, 2021.

[11] A. R. Borah, A. Vijapur, and B. M. Kumar, "Kannada Manuscript Digitization through OCR and Machine Learning," in Proc. 2024 Int. Conf. Comput., Power Commun. Technol. (IC2PCT), Bengaluru, India, 2024; IEEE, pp. 1589–1594

[12] A. S. Menon, R. Sreekumar, and B. N. B. J., "Character and Word Level Recognition from Ancient Manuscripts using Tesseract," in Proc. 2023 Int. Conf. Inventive Comput. Technol. (ICICT), Mysuru, India, 2023; IEEE, pp. 1743–1748

[13] A. Moudgil, S. Singh, B. Sareen, and S. Wadhwa, "Devanagari Ancient Manuscript Recognition Using AlexNet," in Proc. 2024 11th Int. Conf. Reliability, Infocom Technol. Optimization (ICRITO), Noida, India, 2024

[14] T. J. Alexander, S. S. Kumar, and K. Muthuvel, "Performance Evaluation of Pre-processing Techniques for Historical Palm Leaf Manuscript Image Restoration," in Proc. 2020 4th Int. Conf. IoT in Social, Mobile, Analytics and Cloud (I-SMAC), Chennai, India, 2020